SYNTCOMP 2026: Results

The Reactive Synthesis Competition

https://www.syntcomp.org/

Swen Jacobs Guillermo A. Pérez Philipp Schlehuber-Caissier

SYNT@CAV/FLoC 2026

Three messages from 2026

1. The results

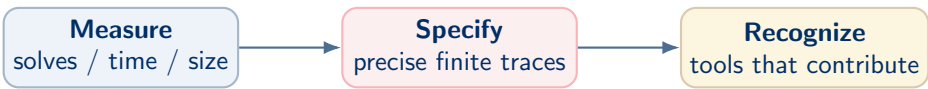
Clear leaders emerge, but coverage, time, and quality still reward different configurations.

2. A firmer standard

LTL_f controllers now have an explicit answer to who stops the trace, how, and when.

3. A broader community

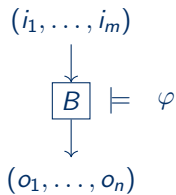
New TLSF tooling, a mailing list, and two contribution medals at the banquet.



Reactive synthesis in one slide

Linear temporal specifications

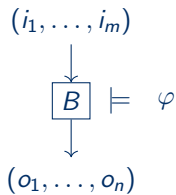
Is there a sequential circuit B that satisfies a given linear temporal logic formula φ on its inputs I and outputs O, for any sequence of input valuations?

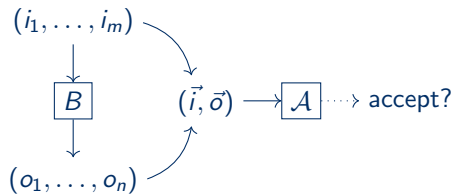


Reactive synthesis in one slide

Linear temporal specifications

Is there a sequential circuit B that satisfies a given linear temporal logic formula φ on its inputs I and outputs O, for any sequence of input valuations?

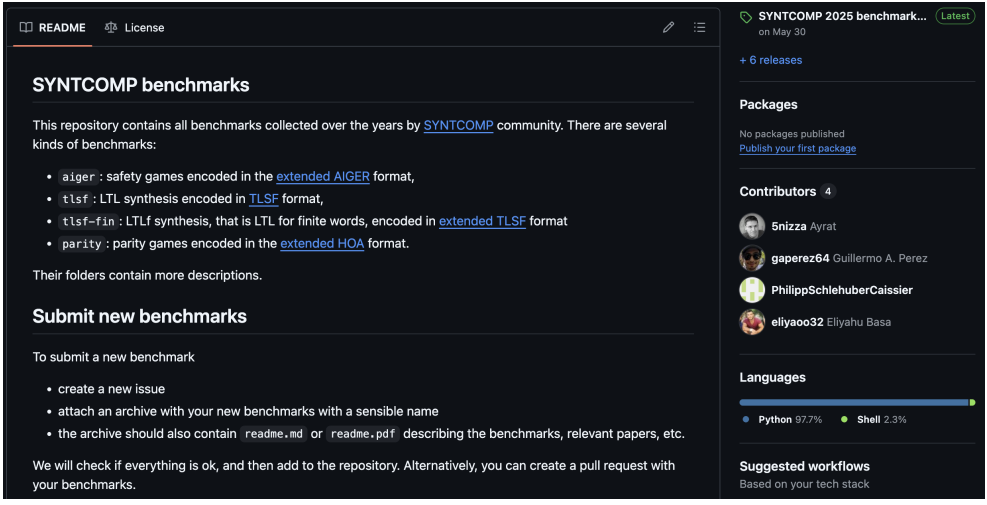




Automata specifications

Is there a circuit B for a given automaton $\mathcal{A}$ such that it accepts, for any sequence of input valuations I?

Benchmarks



README License

SYNTCOMP benchmarks

This repository contains all benchmarks collected over the years by SYNTCOMP community. There are several kinds of benchmarks:

- `aiger` : safety games encoded in the extended AIGER format,
- `tlsf` : LTL synthesis encoded in TLSF format,
- `tlsf-fin` : LTLf synthesis, that is LTL for finite words, encoded in extended TLSF format
- `parity` : parity games encoded in the extended HOA format.

Their folders contain more descriptions.

Submit new benchmarks

To submit a new benchmark

- create a new issue
- attach an archive with your new benchmarks with a sensible name
- the archive should also contain `readme.md` or `readme.pdf` describing the benchmarks, relevant papers, etc.

We will check if everything is ok, and then add to the repository. Alternatively, you can create a pull request with your benchmarks.

SYNTCOMP 2025 benchmark... Latest
on May 30

+ 6 releases

Packages

No packages published
Publish your first package

Contributors 4

5nizza Ayrat
gaperez64 Guillermo A. Perez
PhilippSchlehuberCaissier
eliyaoo32 Eliyahu Basa

Languages

Python 97.7% Shell 2.3%

Suggested workflows

Based on your tech stack

(https://github.com/SYNTCOMP/benchmarks)

Tracks and metrics

Problems

- Realizability: Does a circuit satisfying the specification exist?
- Synthesis: If it exists, construct it in AIGER format.

Specification types

- LTL over infinite words: TLSF
- LTL over finite words: TLSF, with explicit finite-trace stopping semantics
- Parity automata/games: extended HOA

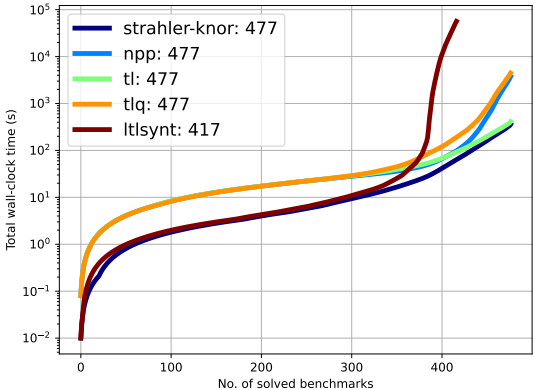
Metrics

Solved instances against wall-clock time, CPU time, memory usage, and for synthesis output size.

2026 tools and data snapshot

Track	Instances	Tool families in the snapshot
Parity-game realizability	483	Knor; `ltlsynt`
LTL$_f$ realizability	488	`ltlfsynt`; Cosy; LydiaSynt
LTL realizability	1524	Acacia-Bonsai; `ltlsynt`; SemML; Strix legacy
LTL synthesis	1505	SemML; Strix legacy; `ltlsynt`

Parity-game realizability: near-complete coverage



Coverage

Four KNOR configurations each solve **477 of 483** instances.

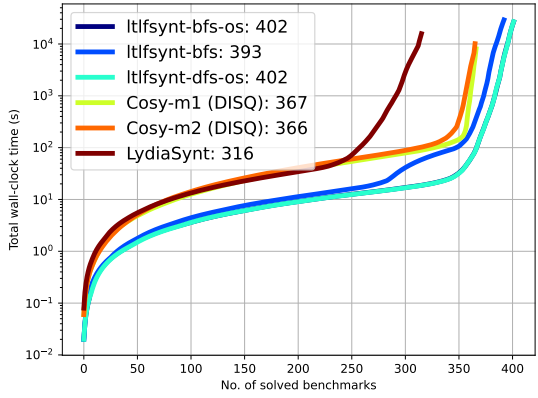
Fastest aggregate

`strahler-knor`: **360 s** cumulative wall time. `tl` is close at 410 s.

Baseline

`ltlsynt` solves 417 instances.

LTL_f realizability: two searches tie



Top coverage

`ltlfsynt-bfs-os` and `ltlfsynt-dfs-os` both solve **402 of 488**.

Close aggregate time

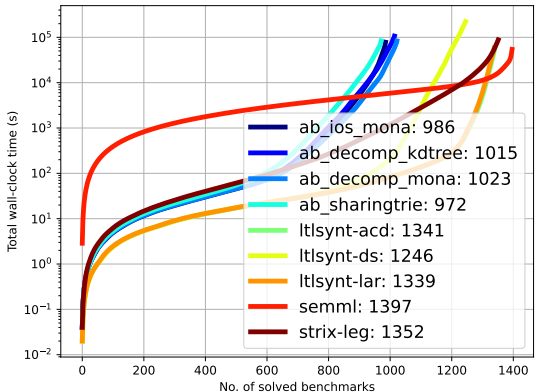
26.74 ks versus 26.95 ks cumulative wall time.

Ranking note

Both Cosy configurations are marked **DISQ** in the supplied data and are excluded from the ranking.

Source: supplied `tlsfFinReal/sumUp.csv` and cumulative wall-clock plot.

LTL realizability: SemML leads coverage



Leader

SemML solves **1397 of 1524** instances.

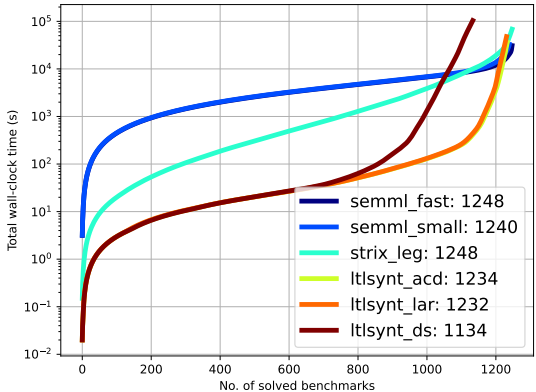
Next group

Strix legacy: 1352
`ltlsynt-acd`: 1341
`ltlsynt-lar`: 1339

Takeaway

SemML adds **45 solves** over the Strix baseline and also has the lowest cumulative wall time in this group.

LTL synthesis: same coverage, different wall time



Coverage tie

`semml_fast` and Strix legacy each solve **1248 of 1505**.

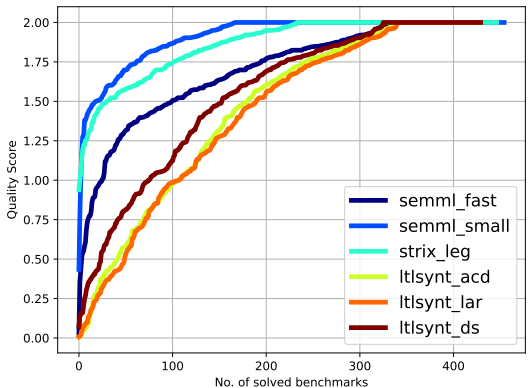
Aggregate time

`semml_fast`: 30.9 ks
Strix legacy: 68.4 ks

Interpretation

At equal coverage, SemML-fast uses **2.2× less** cumulative wall time.

LTL synthesis: quality remains a separate race



Mean quality score

Configuration	Mean
`semml_small`	**1.91**
Strix legacy	1.86
`semml_fast`	1.70
`ltlsynt-ds`	1.53
`ltlsynt-acd`	1.44
`ltlsynt-lar`	1.41

Trade-off

`semml_small` gives the best mean quality while solving 1240 instances—only eight fewer than the maximum coverage.

LTL_f semantics and TLSF tooling

LTL_f: the stopping convention is explicit

The controller decides when the finite trace ends via alive signal

TLSF tools: a lightweight translation toolkit

New tool: `tlsf2ltl` translates TLSF 1.1/1.2 specifications into LTL(f)

Reference specification and semantics paper:

arXiv:2303.03839

Repository and command-line tools:

github.com/gaperez64/tlsf-tools

The SYNTCOMP community channel is live

Subscribe

`lists.uantwerpen.be/.../syntcomp`

New: the SYNTCOMP mailing list

A durable place for calls, solver and benchmark updates, evaluation notices, and result announcements.

Still open after the annual freeze

The rolling setup remains: send improved tools and benchmarks, and the organizers can rerun the evaluation and update the rankings.

Three public entry points

Website	`syntcomp.org`
Benchmarks	`github.com/SYNTCOMP/benchmarks`
Discussion	the new mailing list

Two medals later in FLoC

When and where

Tuesday, 28 July 2026
CAV/FLoC banquet
Pavilhão Carlos Lopes, Lisbon

What is recognized

Two tools will receive medals for contributing to SYNTCOMP.

The important distinction

Contribution $\neq$ track victory

The medals honor contributions that strengthen SYNTCOMP—not necessarily a top-ranked tool.